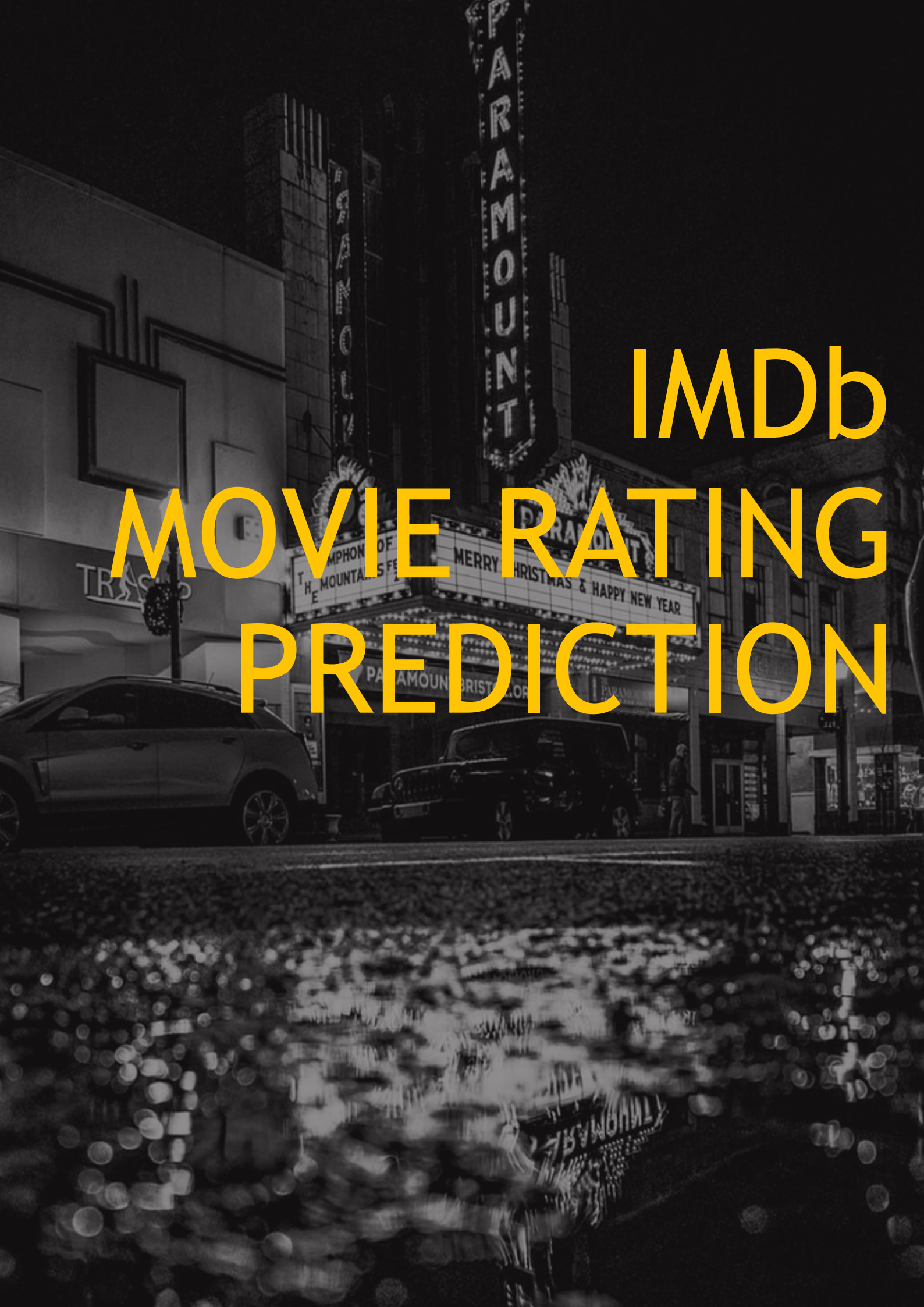
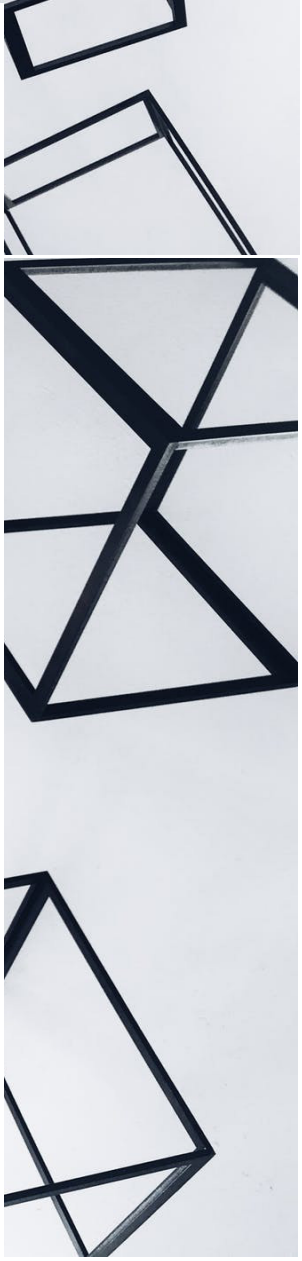



# IMDb MOVIE RATING PREDICTION

# CONTENTS.





# 01 INTRODUCTION

It's a Friday evening and it's the start of the chilly month of November. Our team looks outside through the window and everything is grey and cold. We decide to grab our blanket; it is very cozy and comfy and we decide it's a perfect day to watch a movie. We want to watch a good one, or something at least that it is popular. We have several options, but some of them are not even rated in the typical movie websites! Well, maybe there is a solution for that. Following Spliney's footsteps, we decide to predict a movie's popularity based on certain characteristics…

We have a dataset comprising of 5,456 movies which have been released between 1915 and 2014. The dataset consists of several factors which are linked to the movies. These factors are categorised into 3 major categories:

- **Film Characteristics** - Includes attributes like the budget, the release date, the language most dominant in the films, the genre of the movie, and the total duration of the movie among others.

- **Cast Characteristics** - Includes attributes like the name and the popularity meter of the actors.

- **Production Characteristics** - Includes attributes like the name and the number of directors, producers, editors and production companies involved in the project.

The aim of our project is to filter the relevant predictors to determine the IMDB rating of the upcoming 12 movies released this month.

# 02 DATA DESCRIPTION

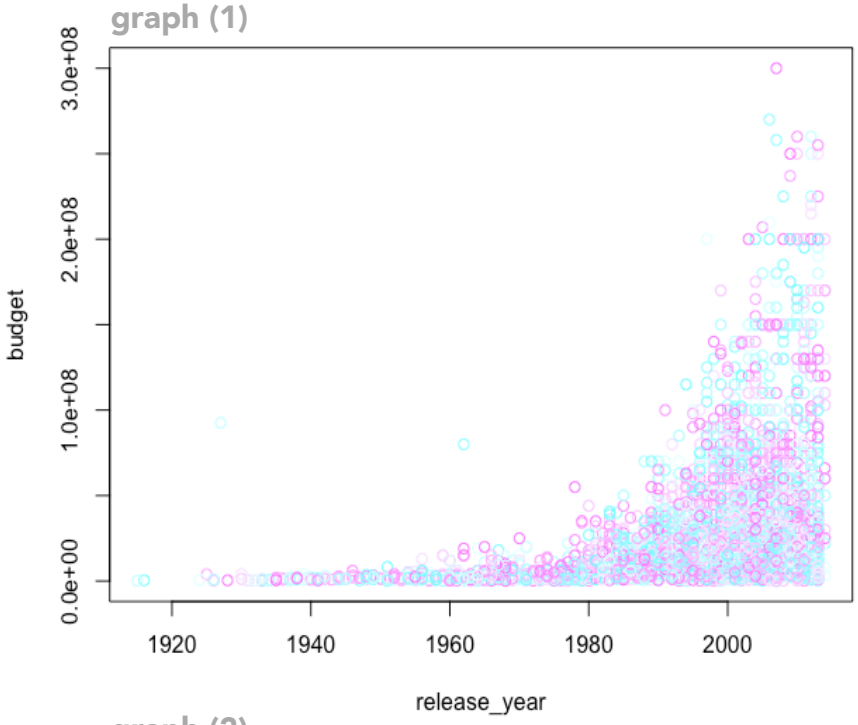
The data given in the raw format needed some preprocessing. The following steps were taken to ready the data for machine learning:

**1.** The "Budget" column had 2,421 null values. Instead of eliminating 2421 rows or the entire budget column, the missing budget values were replaced with the average budget of all movies released in that year. For those years, where no other movie was released the budget was plugged with a value of 0.

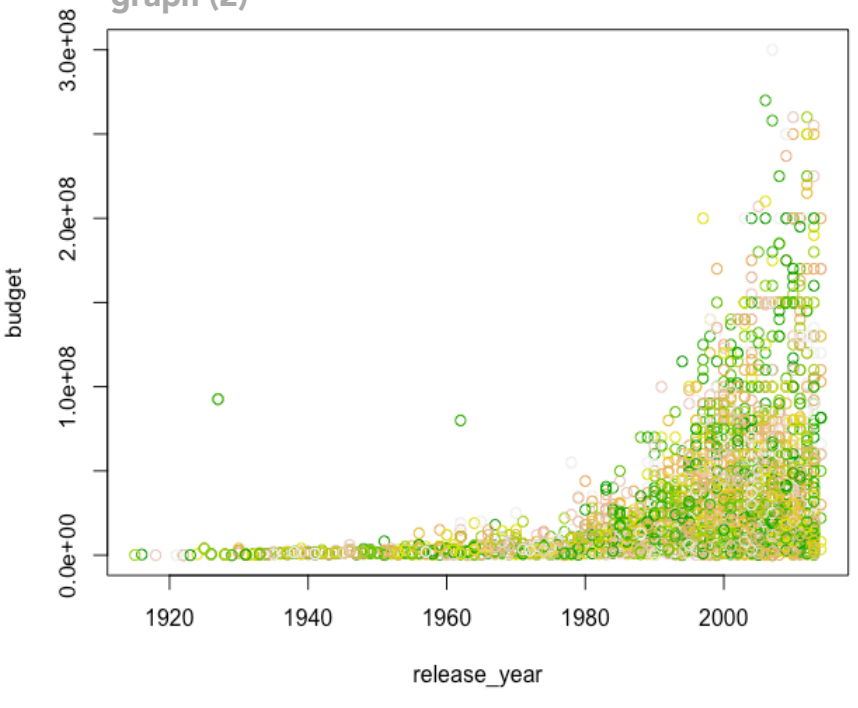
For instance, the budget for the movie "Gunday" is missing. The value is plugged with the average budget of all movies released in 2014 which is 81,617,647. On the other hand, "Orphans of the Storm" is the only movie in the dataset which released in 1921. Therefore, its missing budget is plugged with 0.

Graph (1) shows the values of the "budget" prior to cleaning and preprocessing. Here, we see that movies released from early 1920s to early 1930s do not have the budget value. Graph (2) below shows the budget values plugged in using our preprocessing techniques described above.

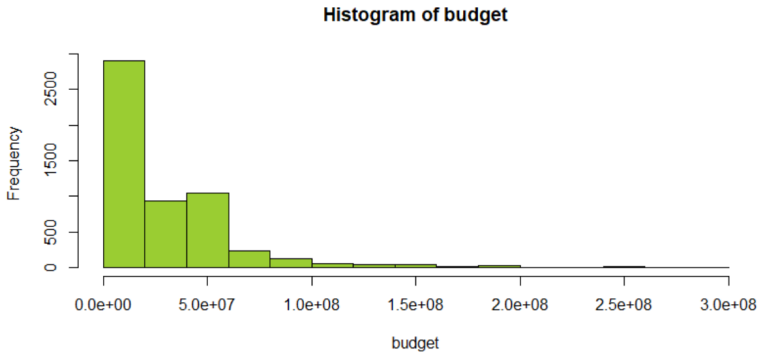
graph (1)



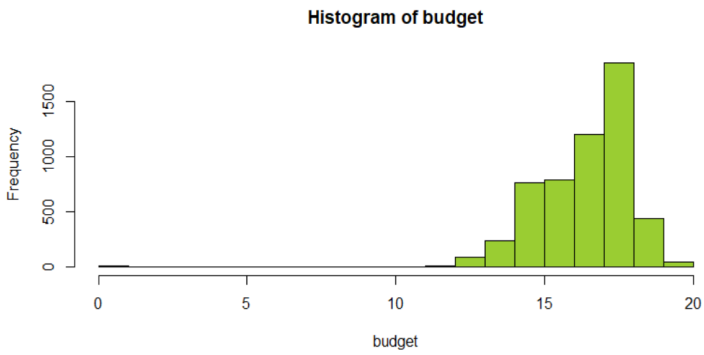

graph (2)



**2.** In graph (3), the "budget" of the movie does not follow a normal distribution. Graph (4) shows the distribution of the budget series after normalization by taking the logarithmic value of the series.

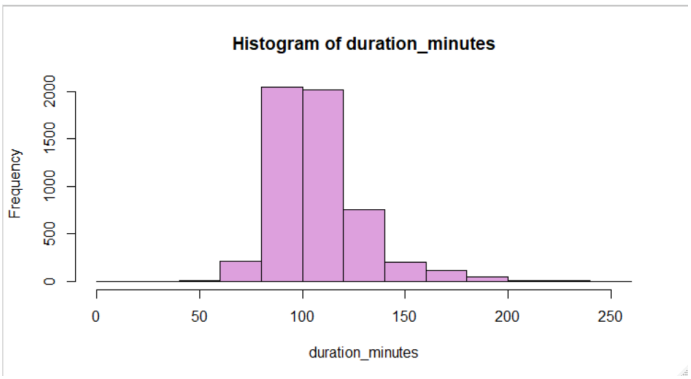



graph (3)




graph (4)

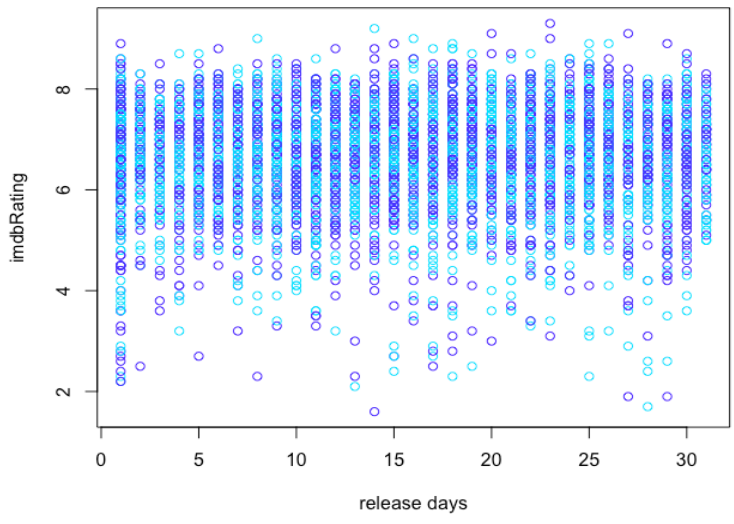
In comparison to "budget", for example, we observe from graph (5) that the data for the "duration_minutes" of the movie is normally distributed and needs no further manipulation.




graph (5)

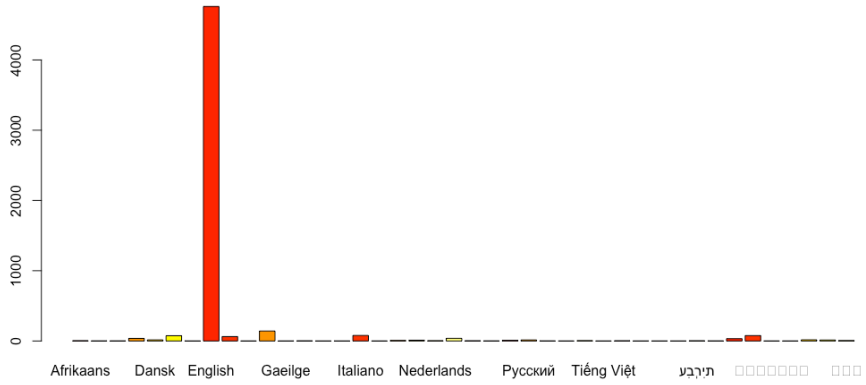
**3.** The column "release_month" has integer values ranging from 1 - 12. These values have been dummified and 11 more columns are created without January to avoid linearity problem.

**4.** Graph (6) demonstrates that there may be no relationship between the "release_day" and the "imdb_rating" and therefore release_day is dropped.




graph (6)

**5.** Graph (7) shows that the main dominant language across most movies is English. Therefore, language will have no impact on the "imdbRating".




graph (7)

**6.** In order to accommodate the impact of the non-numerical attributes like director, producer, editor, production company and production country the following attributes are added:

| Production Characteristics | Total # of the Production Characteristics | # of Production Characteristics Produced 1 Movie |
|---|---|---|
| Director | 2,293 | 1,297 |
| Producer | 2,214 | 1,463 |
| Editor | 1,438 | 695 |
| Production Company | 1,379 | 938 |
| Production Country | 55 | 18 |

table (1)

***i.*** Director_average_rating: This is computed by grouping the directors together and assigning the mean of the imdb_rating based on that directors historical movies to that particular director. Additionally, if the director has directed only one movie (implying that there is no historical data to measure their performance) then in that case we take the director_average_rating as the average imdb_rating for all movies released in that year. Appendix 8 shows the process.
The rationale can be expressed in the following formula:
Let
$n_1$ denote the total number of movies directed by a certain director
m denote the total number of movies released in that specific release year.
i = 1, 2, ,3, ... , $n_1$
j = 1, 2, ,3, ... , m

$$director\_avg\_rating = \begin{cases} \frac{1}{n_1}\sum_{i=1}^{n} ImdbRating_{idirector} \;\; if\; n_1 \geq 1 \\ \frac{1}{m}\sum_{j=1}^{m} ImdbRating_{jrelease\_year} \;\; if\; n_1 = 1 \end{cases}$$

***ii.***The producer_avg_rating, editor_avg_rating, company_avg_rating, production_country_avg_rating are computed using the same logic.

$$producer\_avg\_rating = \begin{cases} \frac{1}{n_2}\sum_{i=1}^{n} ImdbRating_{iproducer} \;\; if\; n_2 \geq 1 \\ \frac{1}{m}\sum_{j=1}^{m} ImdbRating_{jrelease\_year} \;\; if\; n_2 = 1 \end{cases}$$

Let $n_2$ denote the total number of movies produced by a certain producer

$$editor\_avg\_rating = \begin{cases} \frac{1}{n_3}\sum_{i=1}^{n} ImdbRating_{ieditor} \ \ if\ n_3 \geq 1 \\ \frac{1}{m}\sum_{j=1}^{m} Imdb Rating_{jrelease\_year} \ \ if\ n_3 = 1 \end{cases}$$

Let $n_3$ denote the total number of movies edited by a certain editor

$$company\_avg\_rating = \begin{cases} \frac{1}{n_4}\sum_{i=1}^{n} ImdbRating_{icompany} \ \ if\ n_4 \geq 1 \\ \frac{1}{m}\sum_{j=1}^{m} Imdb Rating_{jrelease\_year} \ \ if\ n_4 = 1 \end{cases}$$

Let $n_4$ denote the total number of movies produced by a certain company

$$production\_country\_avg\_rating = \begin{cases} \frac{1}{n_5}\sum_{i=1}^{n} ImdbRating_{icountry} \ \ if\ n_5 \geq 1 \\ \frac{1}{m}\sum_{j=1}^{m} Imdb Rating_{jrelease\_year} \ \ if\ n_5 = 1 \end{cases}$$

Let $n_5$ denote the total number of movies produced by a certain country

**7.** There are 3 types of actors along with their 3 starmeter. These 6 attributes are replaced by the lowest starmeter among the 3 actors. We believe that the most popular actor will be able to drive the popularity and the rating of the movie and the actor#2 and actor#3 columns can be dropped.

**8.** The columns of main_actor_known_for is dropped for all 3 actor types, since the popularity of the actor has already been reflected in main_actor_best_star_meter. Assumption made: If an actor is known for his previous movies then there is no reason to believe that their current movie will have a high imdb_rating. If an actor is not very well known, there is still a probability that their new movie will have a high rating.

**9.** The film_title and url are removed since the name and the url will have no impact on the imdb_ rating.

# 03 MODEL SELECTION

The total_number_of_genres has been eliminated to avoid double counting since each genre is dummified. The correlation matrix below shows the "interaction" between the significant predictors. (Appx 1)

Beginning with the preliminary model (mreg_9), we run a linear regression with ALL the numeric predictors (except for imdb_id) and get a list of 25 significant predictors having p-value less than 0.05. Running a subsequent linear regression with these significant predictor(mreg_10) we further eliminate the total_number_of_production_countries predictor. Testing mreg_10 for outliers we perform the Studentized Outlier test (or the QQ plot) to visually detect outliers. Performing the numerical test for outlier detection we eliminate 10 outliers having p-value less than 0.05 (Appx 2: Table (a)). Eliminating outliers, we run the mreg_11 using the remaining 24 predictors.
Testing for collinearity using the pairs.panels() and the vif() functions we conclude that the correlation coefficients are less than 0.8 and the vif outputs are < 4 and collinearity does not exist in our model. (Appx 2: Table (b)).

We then run the Tukey-test using the ResidualPlots() and conclude that our model is non-linear with 8 non-linear predictors. We run our final model using polynomial regression with 24 predictors. (Appx 3,4) Using K-fold cross validation (K=14) we select the most optimal combination of degrees for our non-linear model which gives the lowest cv error and the best adjusted R-squared (Appx 5). Running the ncvTest, our model is concluded to be heteroskedastic (Table (2)) which is then corrected using the coeftest.

table (2)

| Predictor | Estimate | Std. Error | t value | Pr(>\|t\|) | Significance level |
|---|---|---|---|---|---|
| Intercept | 5.07965 | 0.40351 | 12.589 | < 2e-16 | *** |
| release_year | 6.16921 | 1.08476 | 5.687 | 1.36E-08 | *** |
| release_year2 | 0.33805 | 0.71813 | 0.471 | 0.637843 | |
| release_year3 | -2.66069 | 0.70031 | -3.799 | 0.000147 | *** |
| total_number_of_actors | 6.61579 | 0.69133 | 9.57 | < 2e-16 | *** |
| total_number_of_actors2 | -3.49145 | 0.64861 | -5.383 | 7.64E-08 | *** |
| total_number_of_actors3 | 1.6804 | 0.6435 | 2.611 | 0.009044 | ** |
| duration_minutes | 8.45754 | 0.80513 | 10.505 | < 2e-16 | *** |
| duration_minutes2 | -2.08801 | 0.75862 | -2.752 | 0.005936 | ** |
| duration_minutes3 | 0.38048 | 0.79788 | 0.477 | 0.633474 | |
| director_avg_rating | 24.83302 | 0.87129 | 28.501 | < 2e-16 | *** |
| director_avg_rating2 | -2.64089 | 0.76014 | -3.474 | 0.000516 | *** |
| director_avg_rating3 | -0.14303 | 0.66927 | -0.214 | 0.830776 | |
| company_avg_rating | 9.3756 | 0.73396 | 12.774 | < 2e-16 | *** |
| company_avg_rating2 | -4.67601 | 0.66369 | -7.045 | 2.08E-12 | *** |
| budget | -9.63585 | 1.02539 | -9.397 | < 2e-16 | *** |
| budget2 | -4.29982 | 0.77026 | -5.582 | 2.49E-08 | *** |
| producer_avg_rating | 11.39571 | 0.81816 | 13.928 | < 2e-16 | *** |
| producer_avg_rating2 | 0.25958 | 0.75081 | 0.346 | 0.729553 | |
| editor_avg_rating | 11.07945 | 0.81286 | 13.63 | < 2e-16 | *** |
| editor_avg_rating2 | -1.98043 | 0.70566 | -2.806 | 0.005026 | ** |
| genre_action | -0.10752 | 0.02531 | -4.249 | 2.18E-05 | *** |
| genre_adventure | -0.09834 | 0.02613 | -3.763 | 0.00017 | *** |
| genre_animation | 0.46338 | 0.05206 | 8.901 | < 2e-16 | *** |
| genre_comedy | -0.08272 | 0.02173 | -3.807 | 0.000142 | *** |
| genre_documentary | 0.96141 | 0.08506 | 11.302 | < 2e-16 | *** |
| genre_drama | 0.1061 | 0.02242 | 4.732 | 2.28E-06 | *** |
| genre_family | -0.07528 | 0.03709 | -2.03 | 0.042445 | * |
| genre_fantasy | -0.10866 | 0.03444 | -3.155 | 0.001612 | ** |
| genre_horror | -0.21569 | 0.03484 | -6.19 | 6.44E-10 | *** |
| genre_realitytv | 1.66374 | 0.63963 | 2.601 | 0.009318 | ** |
| genre_scifi | -0.15383 | 0.03358 | -4.581 | 4.72E-06 | *** |
| genre_western | -0.10164 | 0.05225 | -1.945 | 0.051783 | . |
| genre_shortfilm | 0.88948 | 0.28451 | 3.126 | 0.001779 | ** |
| productive_director | 0.17191 | 0.02262 | 7.599 | 3.48E-14 | *** |
| production_country_avg_rating | 0.23891 | 0.06004 | 3.979 | 7.01E-05 | *** |
| main_cast_have_female | -0.12522 | 0.0214 | -5.852 | 5.14E-09 | *** |
| | | | | | |
| Signif. codes: 0'***' 0.001'**' 0.01'*' 0.05'.' 0.1' ' 1 | | | | | |
| Residual standard error: 0.6361 on 5409 degrees of freedom | | | | | |
| Multiple R-squared: 0.5876, Adjusted R-squared: 0.5849 | | | | | |
| F-statistic: 214.1 on 36 and 5409 DF, p-value: < 2.2e-16 | | | | | |

# 04
# RESULTS

We choose the K-fold cross validation with K=14 to balance the accuracy and efficiency while deciding the final model. Table (3) shows the final result of our model as well as the most appropriate degree found for those 8 non-linear predictors selected in the last section.

| Predictor | Estimate | Std. Error | t value | Pr(>\|t\|) | Significance level |
|---|---|---|---|---|---|
| Intercept | 5.079649 | 0.402266 | 12.6276 | < 2.2e-16 | *** |
| release_year | 6.169213 | 1.052038 | 5.8641 | 4.79E-09 | *** |
| release_year2 | 0.338053 | 0.756158 | 0.4471 | 0.6548444 | |
| release_year3 | -2.660685 | 0.778909 | -3.4159 | 0.0006404 | *** |
| total_number_of_actors | 6.615788 | 0.652491 | 10.1393 | < 2.2e-16 | *** |
| total_number_of_actors2 | -3.491452 | 0.412619 | -8.4617 | < 2.2e-16 | *** |
| total_number_of_actors3 | 1.680399 | 0.455072 | 3.6926 | 0.0002242 | *** |
| duration_minutes | 8.457545 | 0.871979 | 9.6992 | < 2.2e-16 | *** |
| duration_minutes2 | -2.088011 | 0.779979 | -2.677 | 0.0074507 | ** |
| duration_minutes3 | 0.380483 | 0.774695 | 0.4911 | 0.6233478 | |
| director_avg_rating | 24.833018 | 0.958181 | 25.9168 | < 2.2e-16 | *** |
| director_avg_rating2 | -2.640894 | 0.997846 | -2.6466 | 0.0081542 | ** |
| director_avg_rating3 | -0.143034 | 0.895691 | -0.1597 | 0.87313 | |
| company_avg_rating | 9.375597 | 0.813606 | 11.5235 | < 2.2e-16 | *** |
| company_avg_rating2 | -4.676009 | 0.80659 | -5.7973 | 7.12E-09 | *** |
| budget | -9.635847 | 1.082978 | -8.8976 | < 2.2e-16 | *** |
| budget2 | -4.29982 | 0.757208 | -5.6785 | 1.43E-08 | *** |
| producer_avg_rating | 11.39571 | 0.928765 | 12.2698 | < 2.2e-16 | *** |
| producer_avg_rating2 | 0.259584 | 1.161263 | 0.2235 | 0.8231266 | |
| editor_avg_rating | 11.079445 | 0.859105 | 12.8965 | < 2.2e-16 | *** |
| editor_avg_rating2 | -1.980427 | 0.937516 | -2.1124 | 0.0346963 | * |
| genre_action | -0.107523 | 0.026238 | -4.098 | 4.23E-05 | *** |
| genre_adventure | -0.098338 | 0.026606 | -3.6961 | 0.0002211 | *** |
| genre_animation | 0.46338 | 0.05155 | 8.989 | < 2.2e-16 | *** |
| genre_comedy | -0.082718 | 0.021954 | -3.7677 | 0.0001665 | *** |
| genre_documentary | 0.961412 | 0.088149 | 10.9066 | < 2.2e-16 | *** |
| genre_drama | 0.106098 | 0.022091 | 4.8028 | 1.61E-06 | *** |
| genre_family | -0.075279 | 0.039073 | -1.9267 | 0.0540752 | . |
| genre_fantasy | -0.108661 | 0.037846 | -2.8711 | 0.0041061 | ** |
| genre_horror | -0.215687 | 0.040206 | -5.3645 | 8.46E-08 | *** |
| genre_realitytv | 1.663741 | 0.073081 | 22.7658 | < 2.2e-16 | *** |
| genre_scifi | -0.153834 | 0.040024 | -3.8435 | 0.0001227 | *** |
| genre_western | -0.101639 | 0.039556 | -2.5695 | 0.0102111 | * |
| genre_shortfilm | 0.889476 | 0.296757 | 2.9973 | 0.002736 | ** |
| productive_director | 0.171912 | 0.026013 | 6.6087 | 4.25E-11 | *** |
| production_country_avg_ratin | 0.238913 | 0.059769 | 3.9973 | 6.49E-05 | *** |
| main_cast_have_female | -0.125225 | 0.021043 | -5.951 | 2.83E-09 | *** |

Signif. codes: 0‘***’ 0.001‘**’ 0.01‘*’ 0.05‘.’ 0.1‘ ’ 1

table (3)

What do all these jargons, MSE, estimate, multiple r-squared mean? Within degree range 2 to 3, the lowest cv.error which is suggested by the MSE we can get between the observed imdbRating and predicted imdbRating from our model is 0.4078872.

In table (2), the column of “estimates” essentially suggests the relationship between the predictors and imdbRating. The intercept suggests a movie will be rated at 5.07965 if it satisfies the following traits: 1) it belongs to none of the genres; 2) just born out of thin air and without any director, editor, producer, company; 3) come from somewhere outside the earth and has no country of origin; 4) unknown year of releasing and no budget; 5) for any reason it has no duration minutes and no casts played in. Positive estimate values indicates there is a positive relationship between the predictor and imdbRating while negative estimate values is the opposite. The accuracy of the model told by multiple r-squared is 0.5876. That is to say the explanation power of our model is 58.76%.

Now we apply this model to predict the imdbRating of the movies on our movie list and pick up the one we want to watch. After feeding the relevant information into the model, we get the predicted imdbRating of each movie as table (4) shows:

| Title | Current Rating | Prediction | SE |
|---|---|---|---|
| **Mickey and the Bear** | 7 | 7.497176 | **0.247184** |
| **Noelle** | N/A | 6.454310 | N/A |
| **Atlantique** | 7.2 | 6.998689 | **0.040526** |
| **Charlie's Angels** | N/A | 5.867118 | N/A |
| **Le Mans' 66** | 7.6 | 7.230221 | **0.136737** |
| **The Good Liar** | N/A | 6.549685 | N/A |
| **The Report** | 7 | 6.919061 | **0.006551** |
| **Waves** | 7.7 | 7.536417 | **0.026759** |
| **21 Bridges** | 6.4 | 6.684413 | **0.080891** |
| **Beautiful Day in the Neighborhood** | 6.9 | 7.573136 | **0.453112** |
| **Dark Waters** | N/A | 7.488404 | N/A |
| **Frozen 2** | N/A | 6.740644 | N/A |
| **MSE** | | | **0.1416800** |

table (4)

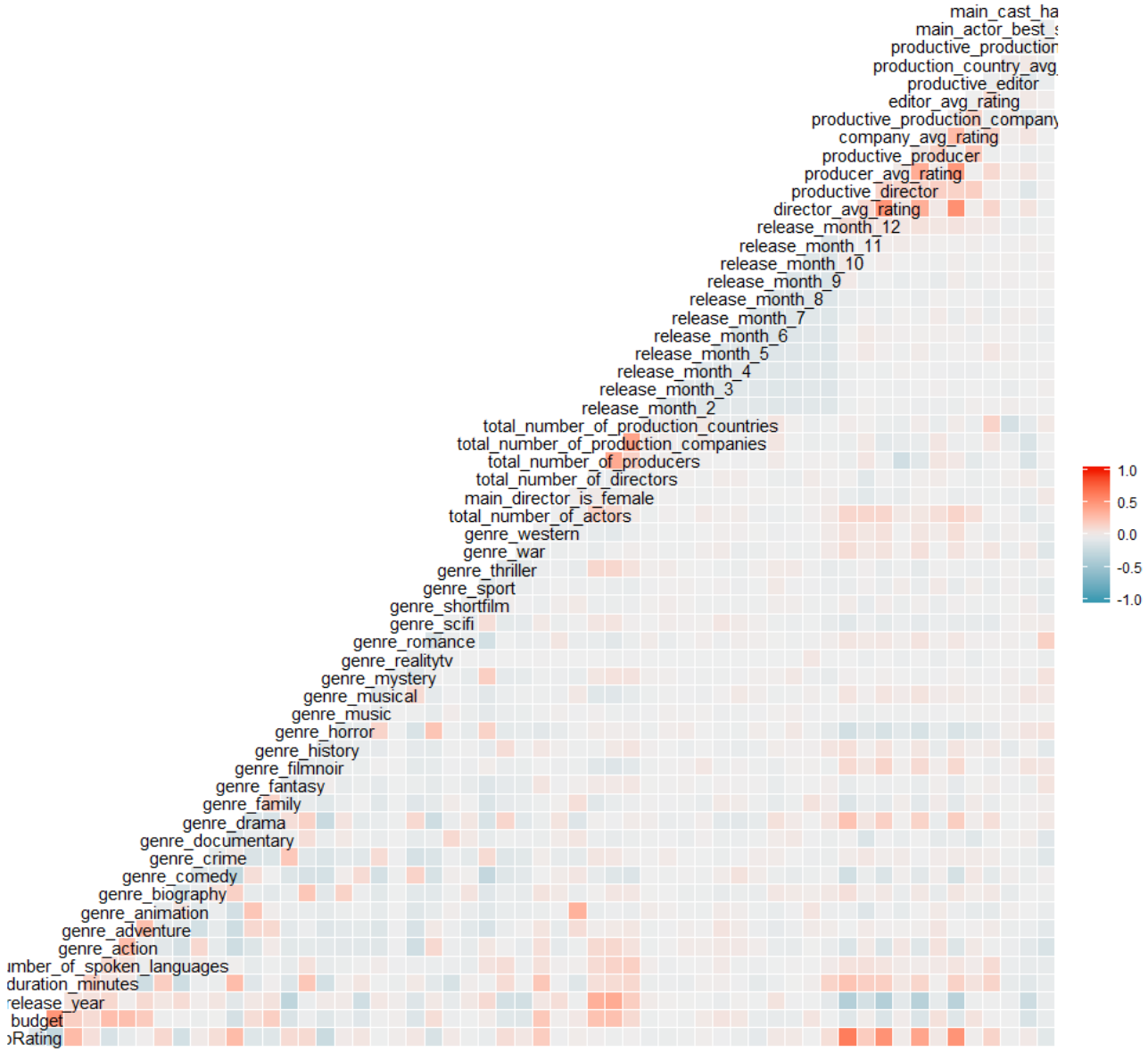
In table (4), "Current Rating" column shows current IMDb rating of some of the movies. "N/A" suggests that there is no rating yet for that particular movie. "Prediction" column releases the rating predicted by our model for each movie. (Briefing the prediction column after the finalized table is post). "SE" reveals the Squared Error between real IMDb rating and predictive IMDb rating. For example, the current rate for Waves is 7.7 and our model gives this movie a 7.5 score; therefore, the squared error is 0.03. Meanwhile, we also calculate the mean squared error which is 0.14 to indicate the performance of the model.

At the end, we would also like to share how we collect information for these 12 targets. Same as problem the original dataset has, there are seven out of 12 movies do not reveal their budget. To solve this problem, we try to find any clues and news related to their budget. For example, some research says Mickey and The Bear is quite a low budget film which always has an average budget at 2.21 million. We also know that Atlantique has the budget of 1.6 million and total number of actors of this movie is nine which only two people more than the amount of actors of Mickey and The bear; therefore, based on this information, we estimate the budget for Mickey and The Bear is one million. We repeat the same process and come up with an estimated budget for Noelle, The Good Liar, The Report, Waves, 21 Bridges, and Frozen 2.

The other highlight is how we collect the average rating for a director, producer, editor, and production company. Take an example of average rating of director. Similar to the method we used to generate the extra column of average rating of director, we used the value in director_avg_rating if the director has already in our table and has directed more than one movie. If the director has already in the table but directed only one movie (implying that there is no historical data to measure their performance) and directors who are not in the table, then in that case we take the director_average_rating as the average IMDB rating of top 10 movies that this director directed. The editor_average_rating, production_company_average_rating, production_country_average_rating are computed using the same logic.

# APPENDIX

# Appendix 1




Correlation Matrix

# Appendix 2

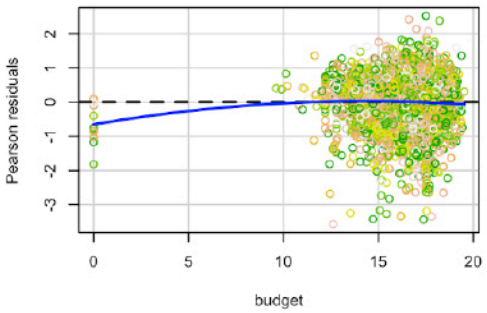
| outlier | rstudent | unadjusted p-value | Bonferroni p |
|---|---|---|---|
| **532** | -7.387488 | 1.72E-13 | 9.40E-10 |
| **104** | -5.911715 | 3.59E-09 | 1.96E-05 |
| **3624** | -5.853092 | 5.11E-09 | 2.79E-05 |
| **4916** | -5.692891 | 1.31E-08 | 7.17E-05 |
| **834** | -5.416397 | 6.34E-08 | 3.46E-04 |
| **4031** | -5.334934 | 9.95E-08 | 5.43E-04 |
| **106** | -5.312127 | 1.13E-07 | 6.15E-04 |
| **200** | -5.241892 | 1.65E-07 | 9.00E-04 |
| **57** | -5.217767 | 1.88E-07 | 1.02E-03 |
| **2429** | -5.18113 | 2.29E-07 | 1.25E-03 |

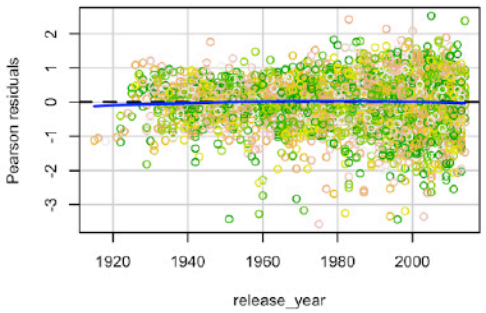
table (a): Outliers

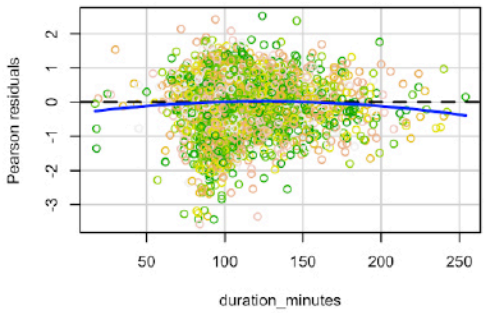
| VIF Test | |
|---|---|
| **budget** | **release_year** |
| 2.326637 | 2.491795 |
| **duration_minutes** | **genre_action** |
| 1.448925 | 1.346657 |
| **genre_adventure** | **genre_animation** |
| 1.269059 | 1.271982 |
| **genre_comedy** | **genre_documentary** |
| 1.426438 | 1.108658 |
| **genre_drama** | **genre_family** |
| 1.668612 | 1.160468 |
| **genre_fantasy** | **genre_horror** |
| 1.092904 | 1.35904 |
| **genre_realitytv** | **genre_scifi** |
| 1.004218 | 1.16742 |
| **genre_western** | **genre_shortfilm** |
| 1.089261 | 1.044629 |
| **total_number_of_actors** | **director_avg_rating** |
| 1.128947 | 1.789733 |
| **productive_director** | **producer_avg_rating** |
| 1.119108 | 1.618679 |
| **company_avg_rating** | **editor_avg_rating** |
| 1.318947 | 1.593684 |
| **main_cast_have_female** | **production_country_avg_rating** |
| 1.066486 | 1.049618 |

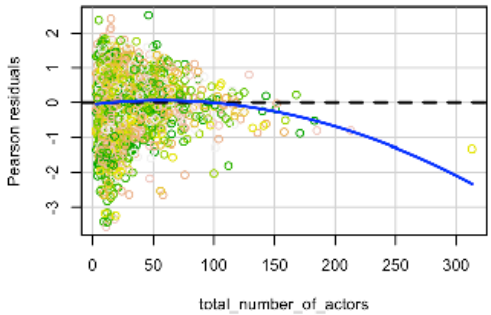
table (b): Collinearity Test

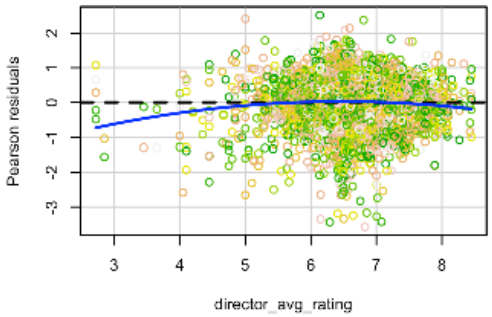
# Appendix 3

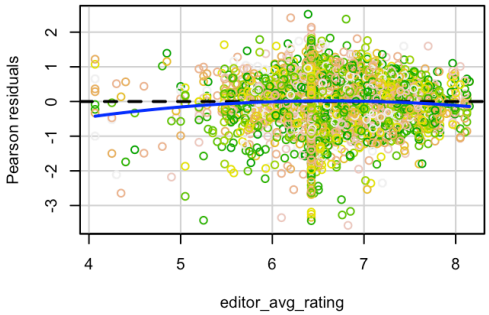


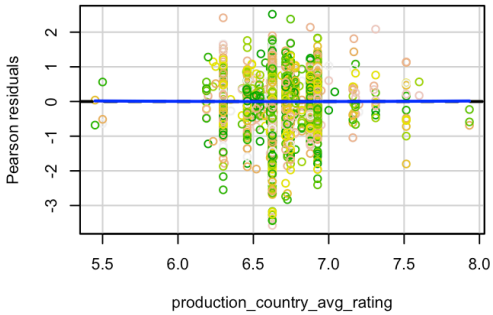



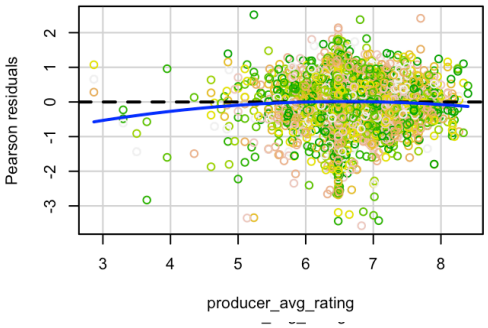



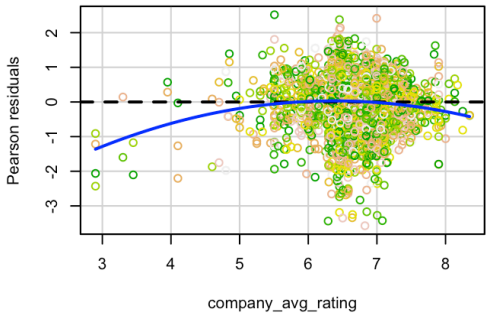



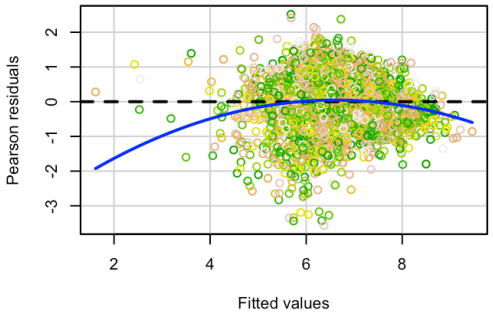




Non-Linearity Test: predictors which are non-linear

# Appendix 4

| Predictor | Test stat | Pr(>\|Test stat\|) | Significance level |
|---|---|---|---|
| budget | -4.5127 | 6.54E-06 | *** |
| release_year | -2.1317 | 0.0330741 | * |
| duration_minutes | -3.4048 | 0.000667 | *** |
| genre_action | -0.6507 | 0.5152448 | |
| genre_adventure | -0.6782 | 0.4976993 | |
| genre_animation | 1.6061 | 0.1083202 | |
| genre_comedy | -1.8788 | 0.0603189 | . |
| genre_documentary | 0.3993 | 0.6896915 | |
| genre_drama | 0.2731 | 0.7848204 | |
| genre_family | -0.6825 | 0.4949346 | |
| genre_fantasy | 0.7602 | 0.4471674 | |
| genre_horror | -0.4411 | 0.6591547 | |
| genre_realitytv | -0.0674 | 0.9462749 | |
| genre_scifi | -0.6951 | 0.4870492 | |
| genre_western | -1.2636 | 0.2064283 | |
| genre_shortfilm | 0.1142 | 0.9091052 | |
| total_number_of_actors | -5.2664 | 1.45E-07 | *** |
| director_avg_rating | -5.5709 | 2.66E-08 | *** |
| productive_director | 1.0867 | 0.2772117 | |
| producer_avg_rating | -3.4282 | 0.0006121 | *** |
| company_avg_rating | -7.6855 | 1.80E-14 | *** |
| editor_avg_rating | -4.3432 | 1.43E-05 | *** |
| production_country_avg_rating | 0.0934 | 0.9255763 | |
| main_cast_have_female | 0.1812 | 0.8562039 | |
| Tukey test | -9.6181 | < 2.2e-16 | *** |
| | | | |
| Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 | | | |

Summary of Non-Linearity Test

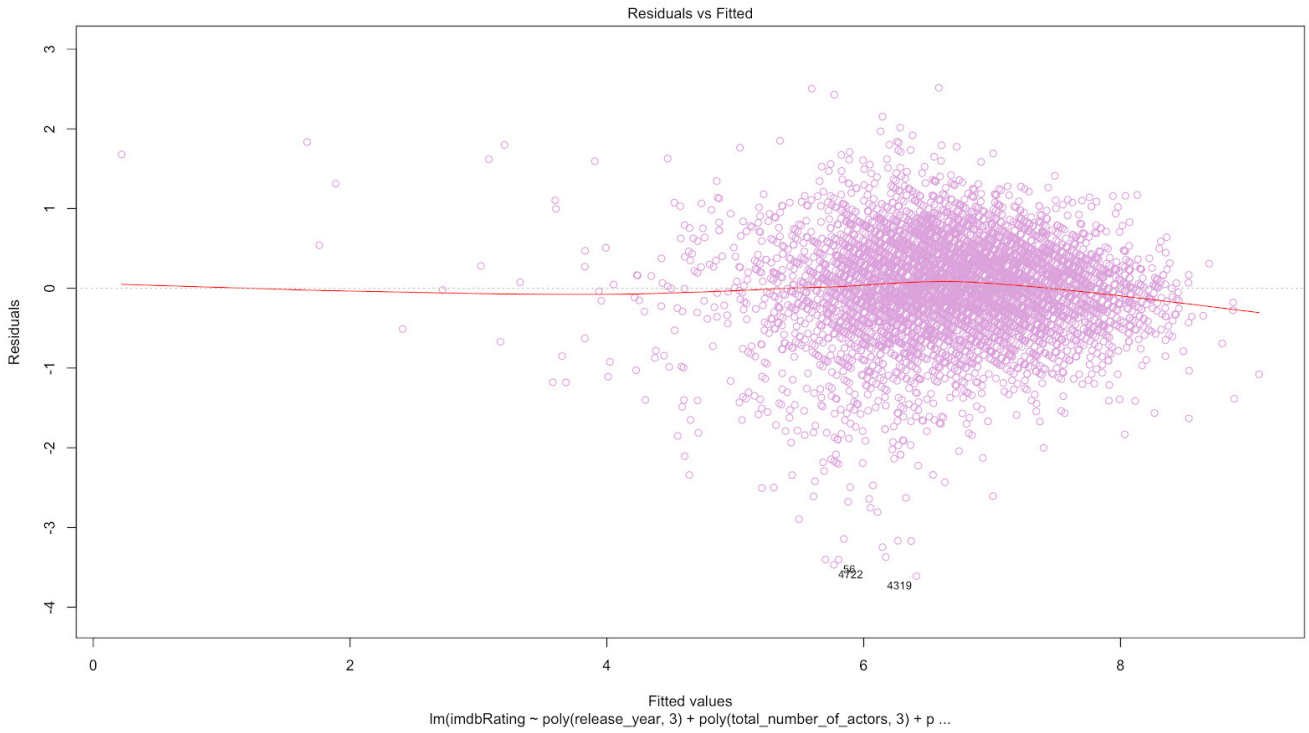
# Appendix 5

| Predictor | Degree |
|---|---|
| budget | 2 |
| release_year | 3 |
| duration_minutes | 3 |
| Total_number_of_actors | 3 |
| director_avg_rating | 3 |
| producer_avg_rating | 2 |
| company_avg_rating | 2 |
| editor_avg_rating | 2 |

Final degrees of non-linear predictors

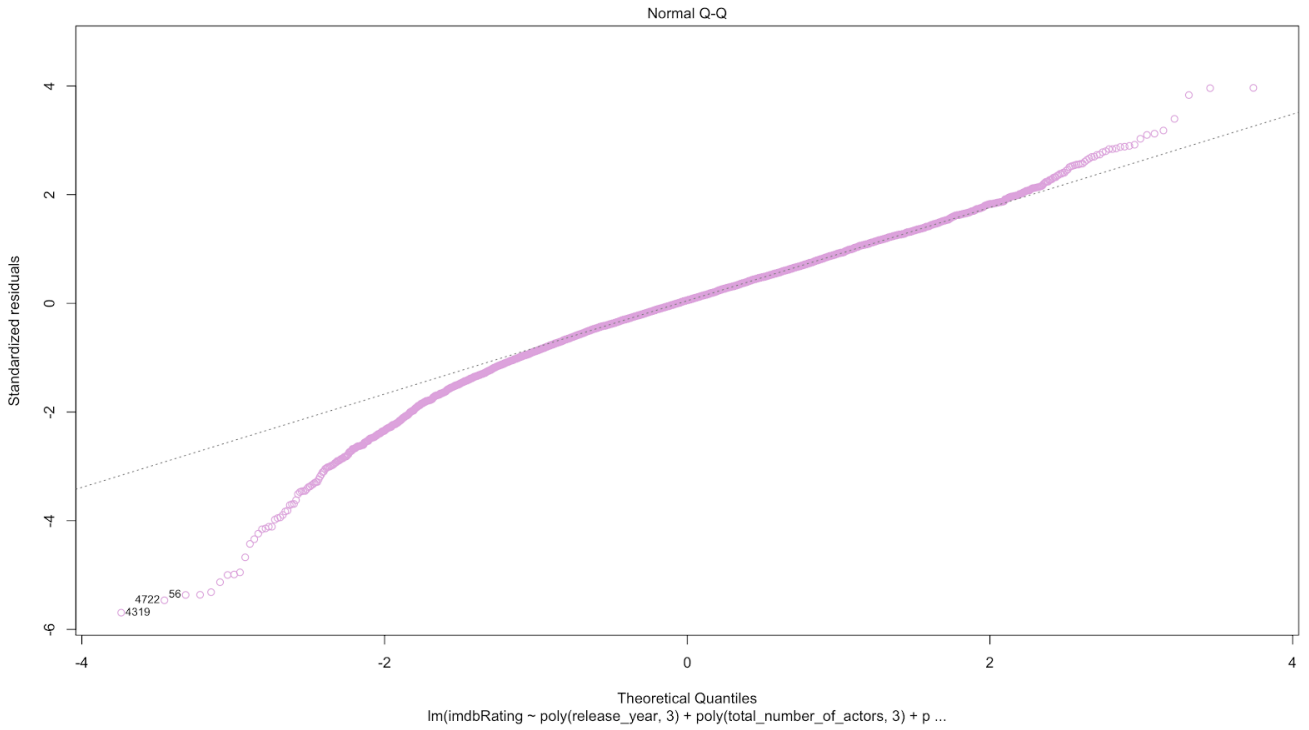
# Appendix 6

The residual plot for the fitted values




graph (a)

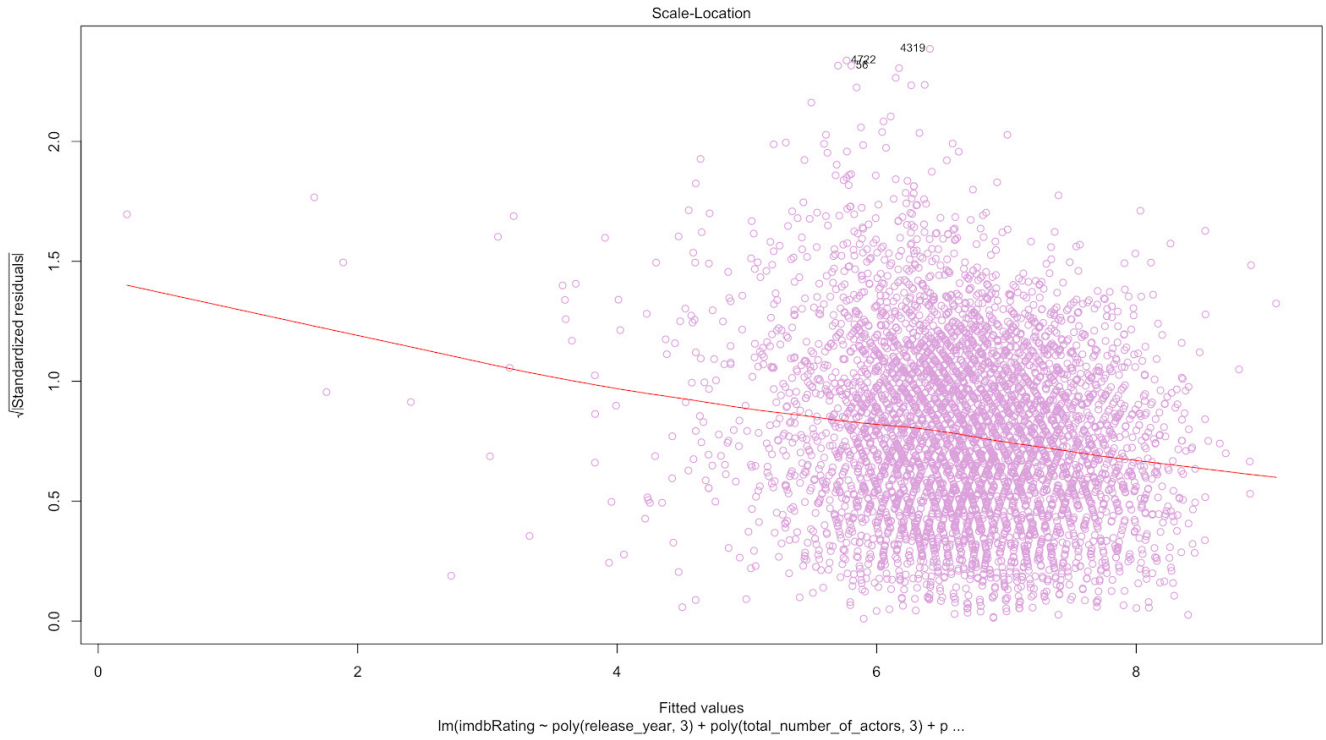
The Normal QQ plot to visually check if the data follows a normal distribution.




graph (b)

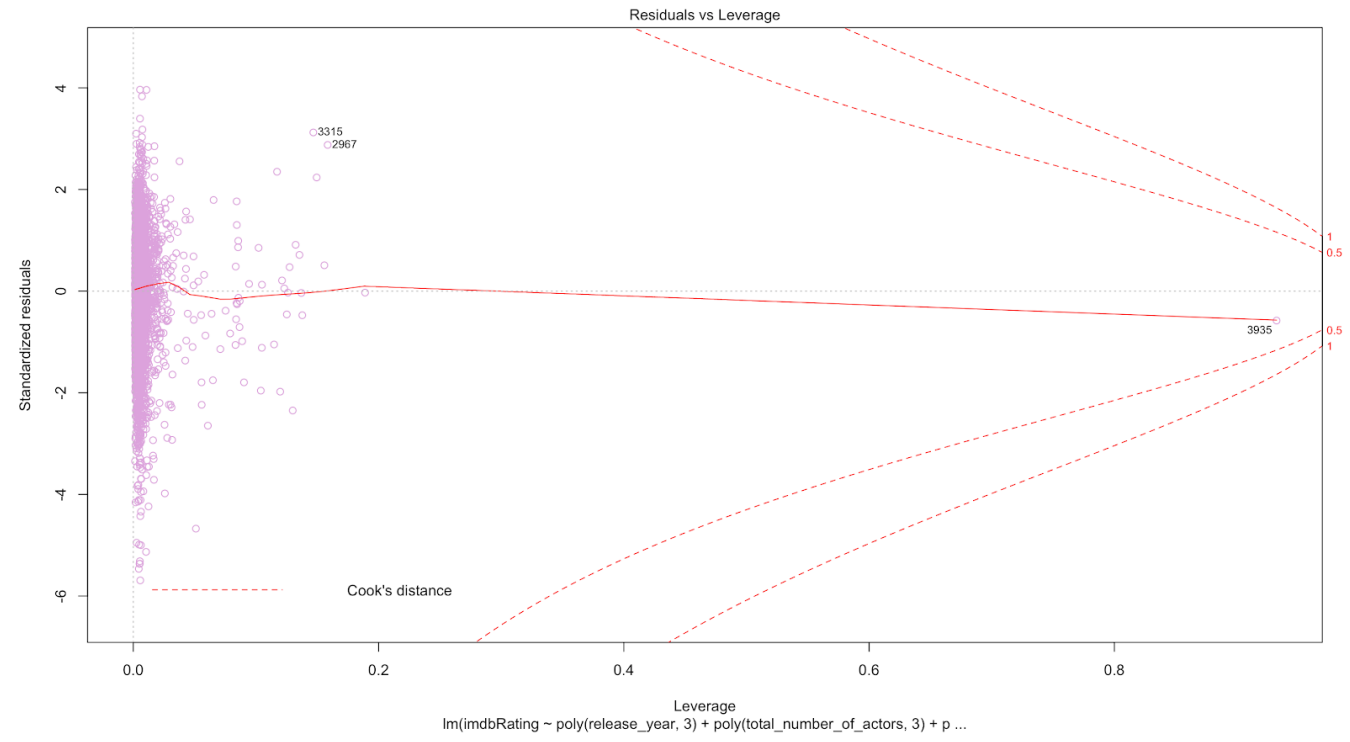
# Appendix 7

The square-root of the absolute standard deviation is plotted against the fitted values of the model.




graph (a)

The residuals is plotted against the average and points below the Cook's distance represent the outliers.




graph (b)

# Appendix 8

| | main_director_name | director_avg_rating | Director_historical_rank |
|---|---|---|---|
| 1 | Vikas Bahl | 9.100000 | 1 |
| 2 | Andrew Kasch | 8.700000 | 2 |
| 3 | Lee Unkrich | 8.500000 | 3 |
| 4 | Rajkumar Hirani | 8.500000 | 4 |
| 5 | Moustapha Akkad | 8.450000 | 5 |

(a)

| | main_director_name | productive_director |
|---|---|---|
| 1 | Aaron Blaise | 0 |
| 2 | Aaron Norris | 0 |
| 3 | Aaron Schneider | 0 |
| 4 | Abbas Alibhai Burmawalla | 0 |
| 5 | Abel Ferrara | 1 |

(b)

Table (a): calculate average imdbRating of each director and rank them
Table (b): differentiate directors with only one movie (0) and directors with more than one movie

| | main_director_name | director_avg_rating | Director_historical_rank | productive_director |
|---|---|---|---|---|
| 1 | Vikas Bahl | 9.100000 | 1 | 0 |
| 2 | Andrew Kasch | 8.700000 | 2 | 0 |
| 3 | Lee Unkrich | 8.500000 | 3 | 0 |
| 4 | Rajkumar Hirani | 8.500000 | 4 | 0 |
| 5 | Moustapha Akkad | 8.450000 | 5 | 1 |

(c)

Table (c): join table (a) and (b)

| release_month_12 | director_avg_rating | Director_historical_rank | productive_director |
|---|---|---|---|
| 0 | 2.500000 | 2283 | 0 |
| 0 | 6.400000 | 1341 | 1 |
| 0 | 6.000000 | 1691 | 0 |
| 0 | 5.000000 | 2125 | 0 |
| 0 | 6.800000 | 957 | 1 |

(d)

Join table (c) with the full dataset and get table (d)

| | release_year | director_yearly_avg |
|---|---|---|
| 1 | 1915 | 6.900000 |
| 2 | 1916 | 8.000000 |
| 3 | 1918 | 6.900000 |
| 4 | 1921 | 8.100000 |
| 5 | 1922 | 7.500000 |

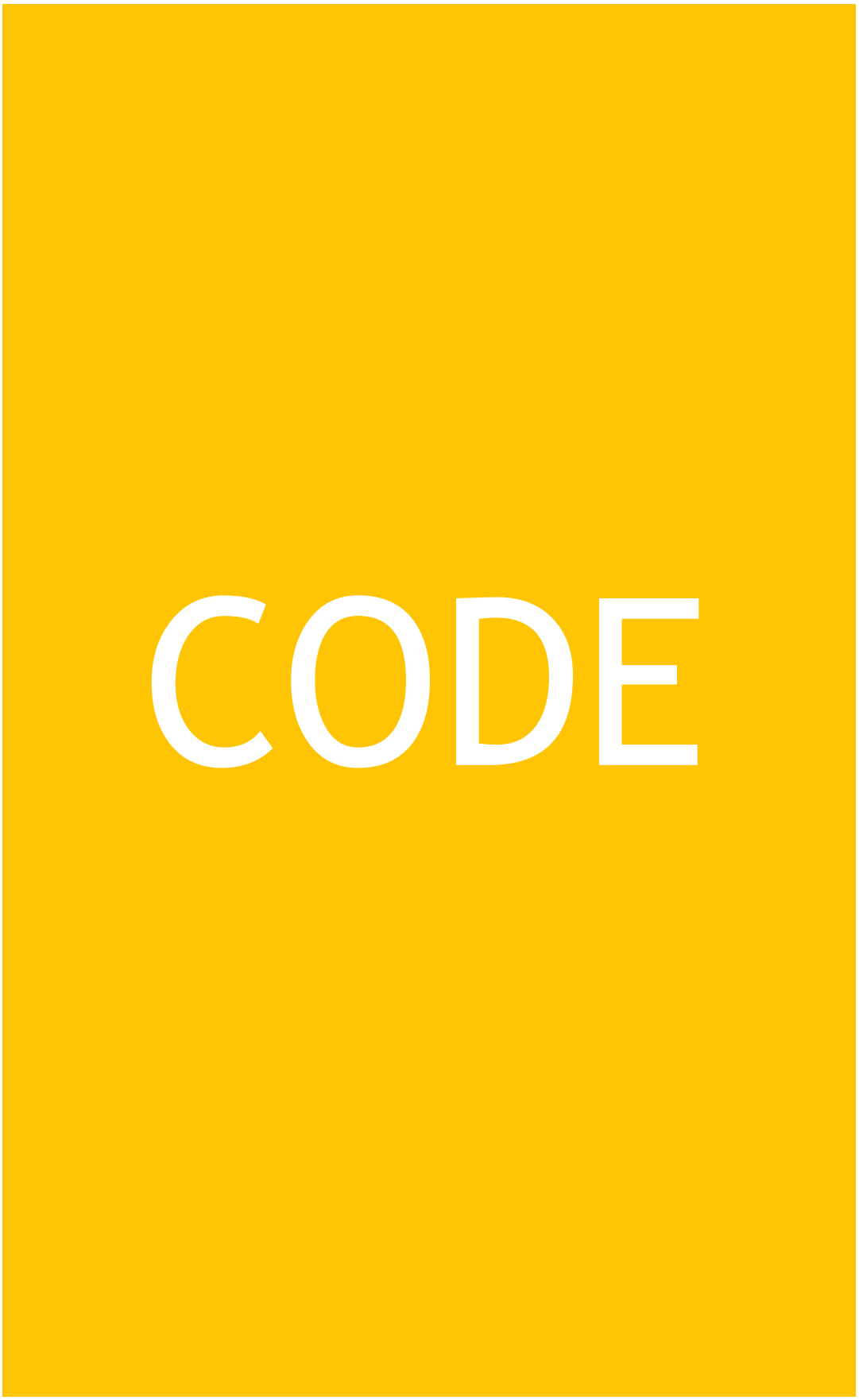
(e)

Table (e): table of imdbRating on a yearly basis

| nth_11 | release_month_12 | director_avg_rating | Director_historical_rank | productive_director |
|---|---|---|---|---|
| | 0 | 6.616667 | 2283 | 0 |
| | 0 | 6.400000 | 1341 | 1 |
| | 0 | 7.005714 | 1691 | 0 |
| | 0 | 6.607914 | 2125 | 0 |

(f)

Join table (e) with the full dataset and table (f)
If “productive_director” = 0, “director_avg_rating” = “director_yearly_rating”


CODE

```
raw_films_dataset_2019 = read.csv('C:\\...\\films_dataset_2019.csv')

attach(raw_films_dataset_2019)
raw_data <- raw_films_dataset_2019

###Step 1: data pre-processing###

###Step 1.1: Film Characteristics pre-processing###

###Descriptive Stats -- Budget###
bdgt_stats <- summary(raw_data$budget)

###Missing values for budget###
attach(raw_data)
plot(release_year,budget,col=cm.colors(10))

###Fill missing budget with yearly average. If no yearly average, replace with 0###
library(dplyr)
raw_data1 <- raw_data %>% group_by(release_year) %>%
   mutate(budget = replace(budget,is.na(budget),mean(budget,na.rm = TRUE))) %>%
   mutate(budget = replace(budget,is.na(budget),0))

attach(raw_data1)
plot(release_year,budget,col=terrain.colors(10))

###Dummify month variables (11 additional columns created without January)###
library(fastDummies)
raw_data2 <- raw_data1 %>% arrange(release_month) %>%
   dummy_cols(select_columns = "release_month", remove_first_dummy = TRUE) %>%
   select(-release_month)

###Reseason why dropping columns of release_day###

plot(release_day,imdbRating,xlab='release days',ylab = 'imdbRating',col=topo.colors(2))

###Below code shows that barely have correlation between release_day and imdbRating###

cor(raw_data2$imdbRating, raw_data2$release_day)
day_avg_rating <- raw_data2 %>% group_by(release_day) %>% summarise(avg_day =
mean(imdbRating))

###The graph shows release_day has no impact on imdbRating###
plot(day_avg_rating$release_day,day_avg_rating$avg_day,col=terrain.colors(10))
raw_data3 <- raw_data2 %>% select(-release_day)

###Reason why dropping columns of main_spoken_language###
plot(raw_data3$main_spoken_language,col=heat.colors(7))
raw_data3$main_spoken_language <- ifelse(raw_data3$main_spoken_language=="English",1,0)
raw_data3 <- raw_data3 %>% select(-main_spoken_language)
```

```
###Step 1.2: Production Characteristics pre-processing###

###Replace director_name with director_avg_rating###

director_reference_table <- raw_data3 %>%
   group_by(main_director_name) %>%
   summarise(director_avg_rating = mean(imdbRating))

director_no.of_film_produced <- raw_data3 %>%
    group_by(main_director_name) %>%
    count(main_director_name) %>%
    rename(productive_director=n) %>%
    mutate(productive_director=ifelse(productive_director>1,1,0))

director_reference_table <- director_reference_table %>%
   left_join(director_no.of_film_produced,by="main_director_name") %>%
   group_by(main_director_name) %>%
    mutate(director_avg_rating=ifelse(productive_director==1,
                           director_avg_rating,mean(director_avg_rating)))
raw_data4 <- raw_data3 %>%
   left_join(director_reference_table,by="main_director_name") %>%
   select(-main_director_name)

director_yearly_mean <- raw_data3 %>% group_by(release_year) %>%
   summarise(director_yearly_avg = mean(imdbRating))

raw_data4 <- raw_data4 %>% left_join(director_yearly_mean,by="release_year") %>%
   mutate(director_avg_rating=ifelse(productive_director==1,
                           director_avg_rating,
                           director_yearly_avg)) %>% select(-director_yearly_avg)

###Replace producer_name with producer_avg_rating###

producer_reference_table <- raw_data4 %>% group_by(main_producer_name) %>%
   summarise(producer_avg_rating=mean(imdbRating))

producer_no.of_film_produced <- raw_data4 %>%
    group_by(main_producer_name) %>%
    count(main_producer_name) %>%
    rename(productive_producer=n) %>%
    mutate(productive_producer=ifelse(productive_producer>1,1,0))

producer_reference_table <- producer_reference_table %>%
    left_join(producer_no.of_film_produced,by="main_producer_name") %>%
    mutate(producer_avg_rating=ifelse(productive_producer==1,
                           producer_avg_rating,
                           mean(producer_avg_rating)))

raw_data5 <- raw_data4 %>% left_join(producer_reference_table,by="main_producer_name") %>%
   select(-main_producer_name)

producer_yearly_mean <- raw_data4 %>% group_by(release_year) %>%
   summarise(producer_yearly_avg = mean(imdbRating))
```

```
raw_data5 <- raw_data5 %>% left_join(producer_yearly_mean,by="release_year") %>%
   mutate(producer_avg_rating=ifelse(productive_producer==1,
                                     producer_avg_rating,
                                     producer_yearly_avg)) %>% select(-producer_yearly_avg)

###Replace production_company with production_company_rating ###

production_company_reference_table <- raw_data5 %>% group_by(main_production_company) %>%
   summarise(company_avg_rating=mean(imdbRating))

production_company_no.of_film_produced <- raw_data5 %>%
   group_by(main_production_company) %>%
   count(main_production_company) %>%
   rename(productive_production_company=n) %>%
   mutate(productive_production_company=ifelse(productive_production_company>1,1,0))

production_company_reference_table <- production_company_reference_table %>%
   left_join(production_company_no.of_film_produced,by="main_production_company") %>%
   mutate(company_avg_rating=ifelse(productive_production_company==1,
                                    company_avg_rating,
                                    mean(company_avg_rating)))

raw_data6 <- raw_data5 %>% left_join(production_company_reference_table,
                                     by="main_production_company") %>%
   select(-main_production_company)

production_company_yearly_mean <- raw_data5 %>% group_by(release_year) %>%
   summarise(company_yearly_avg = mean(imdbRating))

raw_data6 <- raw_data6 %>% left_join(production_company_yearly_mean,by="release_year") %>%
   mutate(company_avg_rating=ifelse(productive_production_company==1,
                                    company_avg_rating,
                                    company_yearly_avg)) %>% select(-company_yearly_avg)

###Replace editor_name with editor_avg_rating###

editor_reference_table <- raw_data6 %>% group_by(editor_name) %>%
   summarise(editor_avg_rating=mean(imdbRating))

editor_no.of_film_produced <- raw_data6 %>%
    group_by(editor_name) %>%
    count(editor_name) %>%
    rename(productive_editor=n) %>%
    mutate(productive_editor=ifelse(productive_editor>1,1,0))

editor_reference_table <- editor_reference_table %>%
    left_join(editor_no.of_film_produced,by="editor_name") %>%
    mutate(editor_avg_rating=ifelse(productive_editor==1,
                                    editor_avg_rating,
                                    mean(editor_avg_rating)))

raw_data7 <- raw_data6 %>% left_join(editor_reference_table,by="editor_name") %>%
   select(-editor_name)

editor_yearly_mean <- raw_data6 %>% group_by(release_year) %>%
   summarise(editor_yearly_avg = mean(imdbRating))
```

```
raw_data7 <- raw_data7 %>% left_join(editor_yearly_mean,by="release_year") %>%
   mutate(editor_avg_rating=ifelse(productive_editor==1,
                                   editor_avg_rating,
                                   editor_yearly_avg)) %>% select(-editor_yearly_avg)

#####Replace main_production_country with production_country_avg_rating ######################
production_country_reference_table <- raw_data7 %>% group_by(main_production_country) %>%
   summarise(production_country_avg_rating=mean(imdbRating))

production_country_no.of_film_produced <- raw_data7 %>%
   group_by(main_production_country) %>%
   count(main_production_country) %>%
   rename(productive_production_country=n) %>%
   mutate(productive_production_country=ifelse(productive_production_country>1,1,0))

production_country_reference_table <- production_country_reference_table %>%
   left_join(production_country_no.of_film_produced,by="main_production_country") %>%
   mutate(production_country_avg_rating=ifelse(productive_production_country==1,
                                       production_country_avg_rating,
                                       mean(production_country_avg_rating)))

raw_data8 <- raw_data7 %>% left_join(production_country_reference_table,
                                by="main_production_country") %>%
   select(-main_production_country)

country_yearly_mean <- raw_data7 %>% group_by(release_year) %>%
   summarise(country_yearly_avg = mean(imdbRating))

raw_data8 <- raw_data8 %>% left_join(country_yearly_mean,by="release_year") %>%
   mutate(production_country_avg_rating=ifelse(productive_production_country==1,
                                   production_country_avg_rating,
                                   country_yearly_avg)) %>% select(-country_yearly_avg)

###Step 1.3: Cast Characteristics pre-processing###

###Replace all 3 main actor names and start meter with main_actor_best_star_meter###

###based on lowest star_meter among the three start_meters###

###Replace if any of the 3 main_actor is female to main_cast_have female###
main_actor_reference_table <- raw_data8 %>% select(imdb_id,imdbRating, main_actor1_name,
                                       main_actor1_is_female,main_actor1_star_meter,
                                       main_actor2_name,main_actor2_is_female,
                                       main_actor2_star_meter,
                                       main_actor3_name,main_actor3_is_female,
                                       main_actor3_star_meter) %>% rowwise() %>%

###Select the best star_meter among the three actors###
   mutate(main_actor_best_star_meter = min(main_actor1_star_meter,
                                   main_actor2_star_meter,
                                   main_actor3_star_meter)) %>%
```

```
###Replace 3 seperate main_actori_is_female to main_cast_have_female###

   mutate(main_cast_have_female = ifelse(sum(main_actor1_is_female,
                                     main_actor2_is_female,
                                     main_actor3_is_female)>=1,1,0))

raw_data9 <- raw_data8 %>% left_join(main_actor_reference_table %>%
                                select(main_actor_best_star_meter,
                                     main_cast_have_female,imdb_id),by="imdb_id") %>%
   select(-main_actor1_name,-main_actor1_is_female,
         -main_actor1_star_meter,-main_actor2_name,
         -main_actor2_is_female,-main_actor2_star_meter,
         -main_actor3_name,-main_actor3_is_female,-main_actor3_star_meter)

###delete 3 columns of main_actor1_known_for###
raw_data10 <- raw_data9 %>% select(-main_actor1_known_for,-main_actor2_known_for,-main_actor3_
known_for)

###delete columns of ###
raw_data_clean <- raw_data10 %>% select(-film_title,-url)


### Step 2: Model Establishment###

###Exclude colinearity between genre###

final_data <- raw_data_clean %>% select(-total_number_of_genres)
attach(final_data)
names(final_data)

###Normalizing the continuous variables###

###Non-normal distribution for budget###

hist(final_data$budget,col='Yellowgreen')
final_data$budget = log(final_data$budget+1)

###Normalised###

hist(final_data$budget,col='Yellowgreen')

###normally distrbuted###

hist(final_data$duration_minutes,col='Plum')

###This dummifies the months by itself###

mreg_8 <- lm(imdbRating ~ .-imdb_id, data = final_data)
summary(mreg_8)
plot(mreg_8,col='thistle') ###GIVES plots for Residuals, Heteroskedasticity and Outliers
```

```
###Running Linear Regression using significant predictors#####

mreg_9 = lm(imdbRating~budget+release_year+duration_minutes+genre_action+genre_adventure
          +genre_animation+genre_comedy+genre_documentary+genre_drama+genre_family
          +genre_fantasy+genre_horror+genre_realitytv+genre_scifi+genre_western
          +genre_shortfilm+total_number_of_actors+total_number_of_production_countries
          +director_avg_rating+productive_director+producer_avg_rating
          +company_avg_rating+editor_avg_rating+production_country_avg_rating
          +main_cast_have_female)

plot(mreg_9,col=terrain.colors(4))


summary(mreg_9)


mreg_10 = lm(imdbRating~budget+release_year+duration_minutes+genre_action+genre_adventure
           +genre_animation+genre_comedy+genre_documentary+genre_drama+genre_family
           +genre_fantasy+genre_horror+genre_realitytv+genre_scifi+genre_western
           +genre_shortfilm+total_number_of_actors
           +director_avg_rating+productive_director+producer_avg_rating
           +company_avg_rating+editor_avg_rating
           +production_country_avg_rating+main_cast_have_female)

plot(mreg_10,col=terrain.colors(10))
summary(mreg_10)

library(car)
residualPlots(mreg_10)
plot(predict(mreg_10), residuals(mreg_10), col="Lavender")
abline(0,0, lty=2)

###Studentized Outlier Visual Detection###

qqPlot(mreg_10, col = cm.colors(10))

###Numerical outlier detection###

###p-value < 0.05 are reported as outliers###

library(car)
outlierTest(mreg_10)

###Create new dataset without the outliers###

final_data_no_out = final_data[-c(532, 104, 3624, 4916, 834, 106, 4031, 200, 57, 2429),]

detach(final_data)
attach(final_data_no_out)
```

```
### Rerun regression with new dataset without the outliers###

mreg_11 = lm(imdbRating~budget+release_year+duration_minutes+genre_action+genre_adventure
          +genre_animation+genre_comedy+genre_documentary+genre_drama+genre_family
          +genre_fantasy+genre_horror+genre_realitytv+genre_scifi+genre_western
          +genre_shortfilm+total_number_of_actors
          +director_avg_rating+productive_director+producer_avg_rating
          +company_avg_rating+editor_avg_rating
          +production_country_avg_rating+main_cast_have_female)
plot(mreg_11,col=heat.colors(4))
summary(mreg_11)

### Collinearity detection###

str(final_data_no_out)
require(psych)
library(ggplot2)
library(GGally)
ggcorr(final_data_no_out)



### Taking only the quantitative variables from the final_data_no_out dataset###

quantvars = final_data_no_out[,c(3,4,5,7,8,9,11,13,14,15,16,19,23,25,26,30,31,48,49,50,52,54,56,59)]
pairs.panels(quantvars)

###Rule of thumb: Collinearity is problematic when absolute value of correlation coefficient

###is above 0.8 or 0.85###

###VIF > 4 is collinearity###

require(car)
vif(mreg_11)

###Recheck nonlinearity###

residualPlots(mreg_11, col = terrain.colors(10))
plot(predict(mreg_11), residuals(mreg_11), col=terrain.colors(10))
abline(0,0, lty=2)


###Nonlinear degree selection (k-fold with polynomial spline)###

require(caTools)
library(boot)
```

```
batman = 5000000 ###Initiate a large number to store the least MSE in the loop###############
joker = c(0,8) ###Initiate an empty vector to store the best combination of degrees##########

for (a in 2:3){
   for (b in 2:3){
      for (c in 2:3){
         for (d in 2:3){
            for (e in 2:3){
               for (f in 2:3) {
                  for (g in 2:3) {
                     for (h in 2:3) {
                        lakers_1=glm(imdbRating~poly(budget,a)
                                    +poly(release_year,b)+poly(duration_minutes,c)
                                    +genre_action+genre_adventure+genre_animation
                                    +genre_comedy+genre_documentary+genre_drama
                                    +genre_family+genre_fantasy+genre_horror
                                    +genre_realitytv+genre_scifi
                                    +genre_western+genre_shortfilm
                                    +poly(total_number_of_actors,d)
                                    +poly(director_avg_rating,e)+productive_director
                                    +poly(producer_avg_rating,f)
                                    +poly(company_avg_rating,g)+poly(editor_avg_rating,h)
                                    +production_country_avg_rating+main_cast_have_female)
                        rocket_1 = cv.glm(final_data_no_out,lakers_1,K=14)$delta[1]
                        if (rocket_1 < batman){
                           batman=rocket_1
                           joker[1]=a
                           joker[2]=b
                           joker[3]=c
                           joker[4]=d
                           joker[5]=e
                           joker[6]=f
                           joker[7]=g
                           joker[8]=h
                     }
                  }
               }
               }
            }
         }
      }
   }
}

batman
joker
```

```
###Final Model###

final_model <- lm(imdbRating~poly(release_year,3)+poly(total_number_of_actors,3)
            +poly(duration_minutes,3)+poly(director_avg_rating,3)+poly(company_avg_rating,2)
            +poly(budget,2)+poly(producer_avg_rating,2)+poly(editor_avg_rating,2)
            +genre_action+genre_adventure+genre_animation+genre_comedy+genre_documentary
            +genre_drama+genre_family+genre_fantasy+genre_horror+genre_realitytv+genre_scifi
            +genre_western+genre_shortfilm+productive_director+production_country_avg_rating
            +main_cast_have_female)

names(final_model$coefficients) = c('Intercept','release_year','release_year2','release_year3'
                                ,'total_number_of_actors','total_number_of_actors2'
                                ,'total_number_of_actors3','duration_minutes','duration_minutes2'
                                ,'duration_minutes3','director_avg_rating','director_avg_rating2'
                                ,'director_avg_rating3','company_avg_rating','company_avg_rating2'
                                ,'budget','budget2','producer_avg_rating','producer_avg_rating2'
                                ,'editor_avg_rating','editor_avg_rating2','genre_action'
                                ,'genre_adventure','genre_animation','genre_comedy'
                                ,'genre_documentary','genre_drama','genre_family','genre_fantasy'
                                ,'genre_horror','genre_realitytv','genre_scifi','genre_western'
                                ,'genre_shortfilm','productive_director'
                                ,'production_country_avg_rating','main_cast_have_female')

summary(final_model)
plot(fitted(final_model),residuals(final_model),col='plum')
abline(0,0,lty=2,col='grey')
plot(final_model, col = 'plum', ps = 300,cex = 2)

###Heteroskedasty correction###

ncvTest(final_model) ###p value less than 0.05 therefore our model shows Heteoskedasticity#####

#To correct heteroskedastic errors, we require two packages
require(lmtest)
require(plm)
coeftest(final_model, vcov=vcovHC(final_model, type="HC1")) ###Heteroskedasticity corrected####
```

```
### Step 3: Prediction ###

Prediction <- read.csv("C:\\...\\Test.csv")
detach(final_data_no_out)
attach(Prediction)

### Normalising the budget in the prediction dataset###

Prediction$budget = log(Prediction$budget+1)

###Removing the title which is irrelevant for the prediction###

data_for_prediction <- Prediction %>% select(-Title)

predicted_imdbrating <- predict(final_model,newdata=data.frame(data_for_prediction))

predicted_imdbrating

write.csv(predicted_imdbrating, "C:\\...\\Prediction.csv")
```